Statistical Language Models for On-line Handwritten Sentence Recognition

Solen Quiniou, Eric Anquetil, Sabine Carbonnel

Statistical Language Models for On-line Handwritten Sentence Recognition

Solen Quiniou Éric Anquetil Sabine Carbonnel

IRISA/INSA
Campus de Beaulieu
35042 Rennes Cedex, France
Solen.Quiniou, Eric.Anquetil, Sabine.Carbonnel@irisa.fr

Abstract

This paper investigates the integration of a statistical language model into an on-line recognition system in order to improve word recognition in the context of handwritten sentences. Two kinds of models have been considered: n-gram and n-class models (with a statistical approach to create word classes). All these models are trained over the Susanne corpus and experiments are carried out on sentences from this corpus which were written by several writers. The use of a statistical language model is shown to improve the word recognition rate and the relative impact of the different language models is compared. Furthermore, we illustrate the interest to define an optimal cooperation between the language model and the recognition system to re-enforce the accuracy of the system.

1. Introduction

The emergence of new devices such as PDA's and Tablet PC's allows users to write larger pieces of texts. Handwriting recognition systems can thus take advantage of the linguistic context of a word to improve their accuracy.

This paper focus on the impact of language models in an on-line handwriting recognition system where word recognition has already been addressed [3].

The technique the most frequently used in handwriting recognition to incorporate linguistic knowledge comes from speech recognition. In this field, statistical language models (often n-gram models) are the most commonly applied [6].

Several works in off-line handwritten sentence recognition make use of language models [8, 14, 13]. The bigram model used in [8] was built on the Lancaster-Oslo/Bergen (LOB) corpus and was shown to decrease the word error rate by 25.7%. In [13] the relative influence of unigram, bigram and trigram language models is investigated. The bigram model was shown to outperform the unigram model while the trigram one didn't lead to further improvements, in terms of perplexity as well as of word recognition rate. Whereas the influence of language models and of word recognition system outputs were the same in these works, the weight of the language model against the recognition system is optimized in [14]. The word error rate is thus decreased by 47.4% with a bigram model and by 54.4% with a trigram one (leading to a 81.8% word recognition rate).

In on-line recognition of handwritten sentences, the use of n-gram language models is quite recent [10, 9]. In [10] a combination of a statistical bi-class model with 500 classes and a syntactical biclass model with 210 POS tags (*Part Of Speech* tags) was used to reorder N-best sentence hypotheses. This model even outperformed a bigram model since respective decrease in word error rates were 33.8% and 32.4% (the word recognition rate for the combined biclass model being 77.5%). Furthermore, this combined model was shown to be more compact than the bigram one. In [9], the use of bigram models (created on different corpuses) is investigated. The bigram model built on the test set achieves a 50% reduction in the word error rate (corresponding to a 85.1% word recognition rate) whereas it leads to a decrease of 15.1% when built on the Susanne corpus. The decrease is lower with the model built on the Susanne corpus since this corpus doesn't fit with the test sentences.

Since we deal with on-line recognition on potentially low memory devices, a particular attention is paid on the size of the models. According to this constraint we study the integration of different language models (n-gram and n-class models) and compare their performance and compactness. We also focus on an optimal cooperation between the language model and the recognition system by highlighting the impact of a language weight.

Section 2 explains the recognition problem in a statistical way while section 3 describes the language modeling used. An overview of the recognition system is given in section 4 and the integration of language models are related in section 5. Finally, section 6 draws some conclusions.

2. Sentence recognition problem

The aim of sentence recognition is to find the most likely sequence of words $\hat{W}$ between candidate sequences W given a signal S (the handwritten sentence to recognize):

$$\hat{W} = \arg\max_{W} p(W|S). \quad (1)$$

By applying a Maximum A Posteriori (MAP) approach, equation 1 can be rewritten as:

$$\hat{W} = \arg\max_{W} p(S|W)\, p(W) \quad (2)$$

where $p(S|W)$ is the a posteriori probability of the signal S for the given sentence W and is estimated by the recognition system often based on HMM's (we call this term *graphical model*); $p(W)$ is the a priori probability of the sequence W, often given by a statistical *language model*.

Since these probabilities are small, their decimal logarithms are used instead. Furthermore, a *language weight* γ (also called *Grammar Scale Factor*) is introduced in order to balance the influence of the language model against the graphical model. Consequently equation 2 becomes:

$$\hat{W} = \arg\max_{W} \log\left[p(S|W)\right] + \gamma \log\left[p(W)\right]. \quad (3)$$

3. Statistical language modeling

Statistical language modeling aims at capturing regularities of a language by use of statistical inference on a corpus of that language [7]. The a priori probability of a sentence $W = w_1^n = w_1 \dots w_n$ of n words is thus given by:

$$p(W) = \prod_{i=1}^{n} p(w_i|h_i) \quad (4)$$

where $h_i = w_1 \dots w_{i-1}$ is called *history* of word i.

The main problem with equation 4 is the high number of histories leading to a tremendous number of probabilities to estimate. Furthermore, most of these probabilities occur too few times to be estimated reliably. A solution to issue this problem is to merge histories in equivalence classes:

$$p(W) = \prod_{i=1}^{n} p(w_i|h_i) = \prod_{i=1}^{n} p(w_i|\Phi_i(h_i)) \quad (5)$$

where $\Phi_i(h_i)$ assigns to history h_i its equivalence class.

There are several techniques to define $\Phi_i(h_i)$, the simplest one being n-gram language models.

3.1. N-gram language models

N-gram language models merge histories ending with the same n-1 words, in equivalence classes:

$$p(W) = \prod_{i=1}^{n} p(w_i|w_{i-n+1}^{i-1}). \quad (6)$$

The probability $p(w_i|w_{i-n+1}^{i-1})$ given by equation 6 is the relative frequency of the sequence w_{i-n+1}^{i} in a corpus:

$$p(w_i|w_{i-n+1}^{i-1}) = \frac{N(w_{i-n+1}^{i})}{w_{i-n+1}^{i-1}} \quad (7)$$

where $N(.)$ stands for the number of occurrences of a certain event. One problem with this approach is that the model fits to the training corpus and probabilities of non-occurring n-grams (i.e. sequences of n words) are estimated to zero.

One solution to issue this is called *smoothing*. It first reduces probabilities of n-grams occurring in the corpus, then redistributes this mass of probabilities among n-grams never encountered. Among different smoothing techniques we chose the Kneser-Ney modified interpolated method, shown in [5] to be very efficient. Nonetheless one limit of this approach is that non-zero probabilities will be assigned to n-grams impossible from a linguistic point of view.

3.2. N-class language models

For their part, n-class models merge words in classes. In that case, the probability of a word is based on its class and on those of the previous words:

$$p(w_i|w_{i-n+1}^{i-1}) = p(w_i|C_i)\, p(C_i|C_{i-n+1}^{i-1}) \quad (8)$$

where $p(w_i|C_i)$ is the probability of the word w_i in its class C_i and $p(C_i|C_{i-n+1}^{i-1})$ is the probability of the class C_i to occur given the history of classes C_{i-n+1}^{i-1}.

There are two main approaches to create word classes. They can correspond to defined categories which are often the grammatical nature of words (i.e. POS tags). Classes can also be created by a statistical approach which merge words that share the same context. We consider only the latter here and classes are created with the incremental version of the Brown algorithm [2].

3.3. Quality of a model

The quality of a language model is measured in terms of *perplexity* (PP) [7]:

$$PP = 2^{H} \quad (9)$$

where $H = \frac{1}{n} \sum_i p(w_i|h_i)$ is an estimation of the entropy of the model (measured over a text). Intuitively the perplexity can be viewed as the average number of words among which w_i has to be chosen knowing history h_i.

The quality of a recognition system is measured in terms of word recognition rate but since no relationship is clearly established between perplexity and word recognition rate [6], the language model with the lowest perplexity doesn't necessarily lead to the highest word recognition rate.

After presenting statistical language modeling, we describe the overall recognition system. We then focus on the integration of word language models into this system.

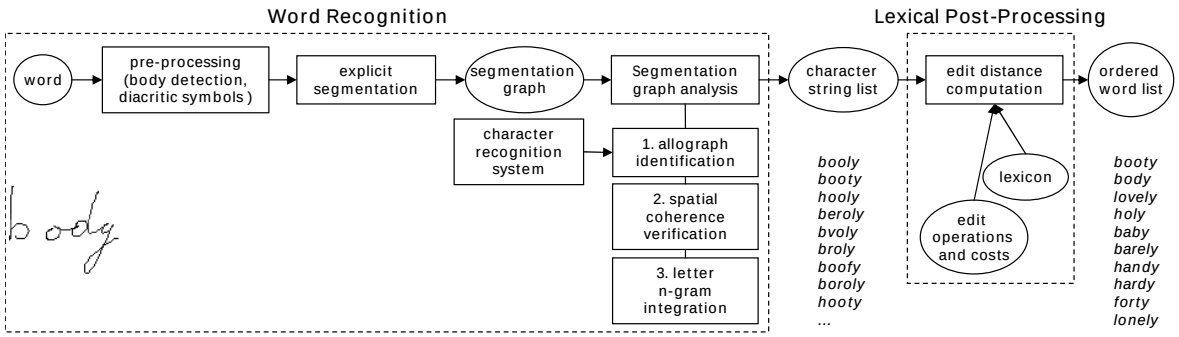

Figure 1. Word recognition system.

4. Overview of the on-line handwriting recognition system

4.1. Word recognition level

Our on-line word recognition system RESIFMot [3] is based on an analytic approach which segments the words according to different hypotheses of letter allographs (see figure 1). The system realizes a physical segmentation based on singularities detection in the signal which provides several logical segmentation hypotheses. Words are then segmented according to different allographs. These allograph hypotheses are organized in a segmentation graph representing all the possible segmentations. An adapted version of our character recognition system RESIFCar [1] is used afterwards to validate the coherent segmentation hypotheses and to identify each allograph hypothesis.

The output of the system is an ordered list of character strings resulting of the segmentation graph exploration. The results are ranked according to the adequation measure between allographs and letter models, the spatial coherence between two allographs and character n-gram probabilities.

The lexical post-processing is finally used to retrieve the nearest word of these results in a dictionary [4]. The output of this post-processing is an ordered list of candidate words.

4.2. Sentence recognition level

Figure 2 describes the structure of the sentence recognition system. Every handwritten word of the segmented sentence is given as an input of the word recognition system which outputs an ordered list of candidate words associated with a likelihood depending on the lexical post-processing. A word graph is then created with these lists. In that graph, each node represents a (n-1)-gram w_{i-n+1}^{i-1} (and is valuated by the likelihood of word w_{i-1}) and each edge from node n_1 to node n_2 (respectively representing (n-1)-grams w_{i-n+1}^{i-1} and w_{i-n+2}^{i}) is valuated by the language model probability $p(w_i|w_{i-n+1}^{i-1})$. This syntactical post-processing step is performed with the Viterbi search algorithm which finds the best path in the graph (corresponding to the likeliest sentence), using word likelihoods and n-gram probabilities as given by equation 3.

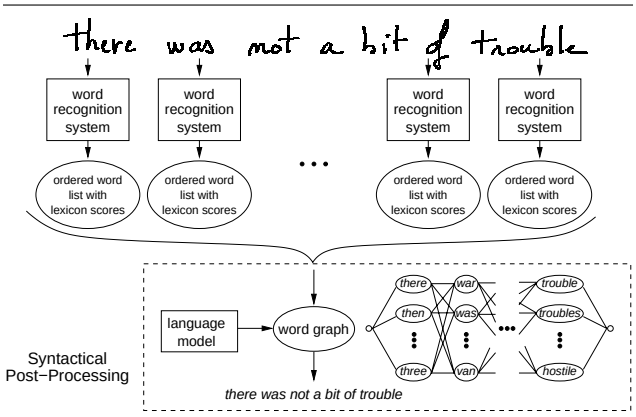

Figure 2. Sentence recognition system.

5. Language model integration: experiments and results

5.1. Corpus and data acquisition

The language models (character and word based) and the lexicon were extracted with the SRILM toolkit [12] from the Susanne corpus [11]. This corpus contains 6,895 sentences (129,460 words); the vocabulary was closed over it and includes 14,392 words.

The handwritten material consists of unconstrained sentences from the Susanne corpus which were written on a Tablet PC by several different writers (figure 3 shows some written words). The sentences were segmented manually into words in order to evaluate the influence of a language model without any bias brought by uncorrect segmentation.

Each writer wrote 20 sentences and shares the first 10 (150 words) with the other writers while the latest 10 were chosen randomly in a set of 109 sentences (1,770 words). The training set includes 138 sentences (2,217 words) written by 7 writers and the test set 80 sentences (1,196 words) written by 4 writers independent from those of the training set. All sentences of both sets were excluded from the corpus used for the construction of the language models.

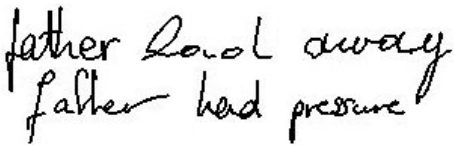

Figure 3. Sample words from the base.

We first present the influence of the language weight when a language model is integrated into the recognition system. Then we compare bigram and trigram models to each other and finally biclass and triclass models as well.

5.2. Influence of the language weight

Equation 3 combines information from word recognition system and language model in the probabilistic case. Since our system is not probabilistic but gives to a handwritten word a likelihood to be the word we want to recognize, we use an approximation of this equation.

For now the optimal value of the language weight γ is set empirically. Figure 4 shows the evolution of the word recognition rate as a function of γ. The two curves (corresponding to bigram and trigram language models) shows a comparable behavior and the optimal value of γ is near 0.3 (the relative impact of both models is not the point here and would be presented in section 5.3). Furthermore, we can see the importance of this weight since the word recognition rate is 86.3 % with the bigram model when γ=1 (graphical and language models have the same impact) whereas it is 90.3 % with the optimal value of γ.

5.3. Bigram model vs trigram model

Here the influence of bigram and trigram models on the recognition is compared to the baseline system (i.e. without language model). Table 1 presents perplexity of both models as well as word recognition rate (also given for the baseline system). The decrease in perplexity from bigram to trigram models is not very important. The same conclusion can be drawn from word recognition rate, showing here a correlation of these two indicators. The fact that the trigram model doesn't perform a significant improvement can be explain by the small amount of trigrams in the test set which are effectively estimated in the model (4.4 %).

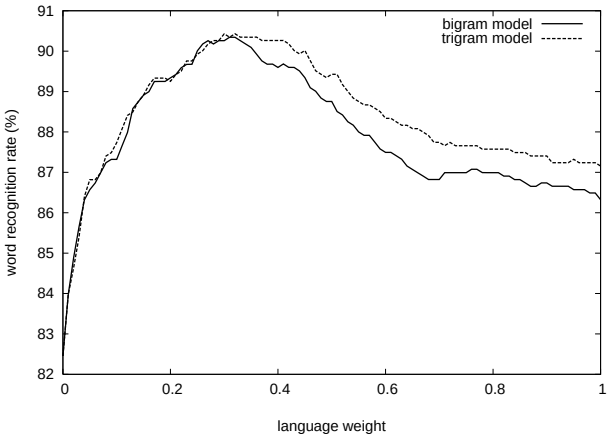

Figure 4. Evolution of the word recognition rate over bigram and trigram models weights.

The influence of a language model is thus shown since the reduction of word error rate is 44.6 % for the bigram model and 45.1 % for the trigram one (compared to the baseline system). In fact, words whose benefit is the highest are small functional words (e.g. *'for', 'to', 'the'*...).

Table 1. Comparison of bigram and trigram models.

	Bigram	Trigram	Baseline
Perplexity	279.4	268.5	-
Word rec. rate	90.3 %	90.4 %	82.5 %

5.4. N-class models: optimizing the number of classes

We investigated here the use of statistical classes. First we focus on the optimal number of classes and then we compare these models to n-gram models, especially in terms of numbers of parameters.

Figure 5 gives the word recognition rate for biclass and triclass models considering different number of classes (the rate for bigram and trigram models are also reminded). We can see that even with 50 classes, the word recognition rate is 88.2 % which is pretty close to the rate obtained with the bigram model. One interesting observation is that the biclass model with 400 classes achieves the same recognition rate than the bigram model (the same conclusion can be drawn from the triclass model with 400 classes and the trigram one). One explanation of that lies in the generalization power of word class clustering. This is particularly beneficial when the amount of training data is too small to estimate reliably n-grams probabilities.

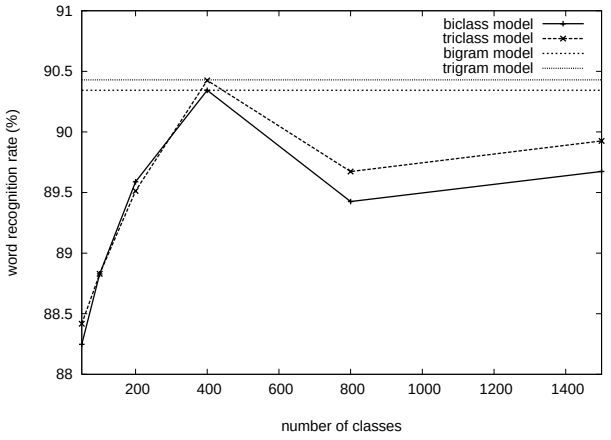

Figure 5. Evolution of the word recognition rate over the number of classes.

Apart from its generalization power, one advantage of n-class models is their compacity. Because words are grouped in classes, the number of n-class probabilities is lower than those of n-grams. These probabilities correspond to the *parameters* of the system (in the case of n-class models, these parameters also include probabilities of words in their classes). Table 2 shows the word recognition rate and the number of parameters for some biclass and triclass models and for the bigram and trigram ones. As can be seen, for the same word recognition rate the biclass model with 400 classes has twice as less parameters as the bigram model (so is the triclass model over the trigram one). Furthermore, with about 16,000 parameters the biclass model with 50 classes achieves an already good word recognition rate.

Table 2. Comparison of biclass, triclass, bigram and trigram models.

	Biclass (50)	Biclass (400)	Triclass (400)	Bigram	Trigram
Word rec. rate	88.2 %	90.3 %	90.4 %	90.3 %	90.4 %
Nb. of param.	16,105	41,112	54,335	86,054	91,882

6. Conclusion

This paper investigates the integration of a statistical language model in a handwritten sentence recognition system. Considering language models built on the Susanne corpus which is relatively small (6,895 sentences and 129,460 words), the results of the experiments show that while a bigram model significantly decreases the word error rate, no further improvements are obtained with a trigram model. We also addresses the use of statistical biclass and triclass models which are interesting because of their compactness.

An optimal language weight was shown to be important. For now this weight is chosen empirically but further work will concern a better combination between the recognition system and the language model based on an automatic supervised learning. Moreover other information provided by the word recognition system could also be integrated in the computation of word likelihoods (e.g. adequation between the word and its graphical shape). Concerning n-class models, only a statistical approach for their creation has been considered and the use of syntactical classes will be explored as well as the combination of both types of classes. Finally, it will be interesting to evaluate the impact of models built on larger corpuses on the recognition accuracy.

References

[1] E. Anquetil and H. Bouchereau. Integration of an On-line Handwriting Recognition System in a Smart Phone Device. In *16th ICPR*, pages 192–195, 2002.

[2] P. Brown, V. D. Pietra, P. de Souza, and J. Lai. Class-Based N-Gram Models of Natural Language. *Computational Linguistics*, 18(4):467–479, 1992.

[3] S. Carbonnel and E. Anquetil. Lexical Post-Processing Optimization for Handwritten Word Recognition. In *7th ICDAR*, pages 477–481, 2003.

[4] S. Carbonnel and E. Anquetil. Lexicon Organization and String Edit Distance Learning for Lexical Post-Processing in Handwriting Recognition. In *9th IWFHR*, pages 462–467, 2004.

[5] S. Chen and J. Goodman. An Empirical Study of Smoothing Techniques for Language Modeling. Technical Report TR-10-98, Harvard University, 1998.

[6] F. Jelinek. *Statistical Methods for Speech Recognition*. MIT Press, 1998.

[7] C. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

[8] U.-V. Marti and H. Bunke. Using a Statistical Language Model to Improve the Performance of an HMM-Based Cursive Handwriting Recognition System. *IJPRAI*, 15(1):65–90, 2001.

[9] S. Marukatat. Sentence Recognition through Hybrid Neuro-Markovian Modeling. In *6th ICDAR*, pages 731–737, 2001.

[10] F. Perraud, C. Viard-Gaudin, E. Morin, and P.-M. Lallican. N-Gram and N-Class Models for On Line Handwriting Recognition. In *7th ICDAR*, pages 1053–1059, 2003.

[11] G. Sampson. *English for the Computer: The Susanne Corpus and Analytic Scheme*. Clarendon Press, 1995.

[12] A. Stolcke. SRILM - An Extensible Language Modeling Toolkit. In *7th ICSLP*, pages 901–904, 2002.

[13] A. Vinciarelli, S. Bengio, and H. Bunke. Offline Recognition of Unconstrained Handwritten Texts using HMMs and Statistical Language Models. *IEEE Transactions on PAMI*, 26(6):709–720, 2004.

[14] M. Zimmermann and H. Bunke. N-Gram Language Models for Offline Handwritten Text Recognition. In *9th IWFHR*, pages 203–208, 2004.